ANONYMOUS AND MALICIOUS

Andreas Hirt and John Aycock
Department of Computer Science, University of Calgary, Calgary, Alberta, Canada

Email hirt@cpsc.ucalgary.ca,
aycock@cpsc.ucalgary.ca

ABSTRACT

Zombie networks have been used for spamming and DDoS attacks. Worms have been designed to receive commands from their creator and update themselves automatically. But the combination of malware and powerful anonymous communication techniques has not been seen – yet.

There is a growing body of research work on anonymous communication schemes, which are developed legitimately to allow people to communicate without fear of identification or retribution. For example, such communication could be used by people living under oppressive regimes.

Malware using anonymous communication would be as capable as current malware ‘applications’, but in a form that is extremely difficult to trace. There are other possibilities, too. An anonymous communication network established using malware could be used for exchanging illegal or copyrighted information, as well as illicit communication for organized crime or terrorist organizations.

This paper discusses anonymous communication methods and shows how they can be modified for use with malware. To counter this threat, we present new methods to identify the existence of malware using anonymous communication schemes, and counterattack techniques that can be used to identify additional nodes within the anonymity network. The awareness of these threats and their countermeasures can be used to build defences before such threats are seen in the wild.

1. INTRODUCTION

How well malware hides is critical to its survival in the wild. Malware may hide in a variety of ways, both active and passive, to obscure its presence or its function. For example, stealth or polymorphism can be used to attempt to evade anti-virus software; captured malware can try to frustrate analysis through anti-debugging. More powerful methods can even make captured malware as hard to directly analyse as breaking strong encryption [9]. (*In this paper, ‘encryption’ refers to strong encryption, not the weak ‘encryption’ sometimes used by malware for obfuscation.*) Successfully hiding increases the time window that the malware has to spread and inflict any damage, whether the damage is incidental or intentional. Malware may also use encryption techniques to hide any mined confidential data, such as credit card numbers that it reports back to its author. It is very difficult for an application that monitors outbound traffic to filter out malware traffic that contains encrypted mined data.

A natural extension for malware is to not only hide itself or its data, but to hide the fact that data is being exchanged at all – we call this *anonymous malware*. From the perspective of the malware author, anonymous communication would have two significant benefits. First, anonymous malware could propagate cautiously by using anonymous communication in order to obfuscate detection via traffic analysis. As a result, it spreads more covertly and it is even more difficult to trace the outbreak to its origin point in order to identify and prosecute its author. Second, the malware could create an anonymous network of corrupted machines that could be used to exchange illegal or copyrighted information, or co-ordinate organized crime or terrorist activity.

The implications of combining malware with anonymous communication are potent. Leveraging research in anonymous communication schemes for malicious purposes can provide a much stronger form of anonymity than controlling botnets via IRC or relaying connections through multiple proxies.

2. ANONYMOUS COMMUNICATION BACKGROUND

The need for legitimate anonymous communication is motivated by a variety of applications, such as:

Whistle blowing. Whistle blowers can perform the useful service of reporting corruption. However, the whistle blower requires anonymity, otherwise retribution from the guilty party could cost the whistle blower their reputation and/or job. For example, Rehan Mullick was the whistle blower for the Iraq Oil-for-Food program [16]. After the whistle blowing, he was repeatedly demoted, and finally terminated by the United Nations in 2001.

Electronic counselling. Anonymity for a victim of abuse is especially important. In the initial stages of recovery, a victim of abuse blames themself for the abuse. The victim does not want to seek help, not only because of fear of retribution from their abuser, but also because of embarrassment. Anonymous communication could provide the necessary support required to overcome this hurdle, allowing the victim to begin the recovery process.

Electronic crime tips. Obviously, there is a very real danger posed to people submitting tips about crimes to police, and complete anonymity is vital.

Military communication. Military communication on the battlefield requires anonymity, so an enemy cannot glean any information from traffic patterns.

Freedom of speech. Under an oppressive regime, electronic voting and even basic communication require anonymity because of possible government retribution.

The two main types of anonymity are data anonymity and connection anonymity [6]. Data anonymity removes identifying data at the application layer, like removing the sender address from an email. Connection anonymity obfuscates traffic patterns to prevent an adversary from identifying the sender and/or receiver of a message. If the anonymous malware exchanges encrypted messages, data anonymity is preserved as long as the decryption key is kept secret. However, despite the use of the encryption, traffic analysis can be performed at all nodes involved in an anonymous malware network, in order to identify additional nodes that are infected. As a result, we only consider connection anonymity in this paper.

Connection anonymity is divided into four types: sender anonymity, receiver anonymity, mutual anonymity, and unlinkability. Sender anonymity [18] is when the sender of a message remains anonymous; receiver anonymity [18] is when the receiver of a message remains anonymous. Mutual anonymity [11] is when both the sender and receiver remain anonymous. Unlinkability [18] is when the sender of a message can be identified, but cannot be linked to the receiver. (Alternately, the receiver of a message could be identified, but without being linked to the sender.)

To obtain anonymity, the identity of the sender or receiver of a message is hidden in a subset of the participants. The sender anonymity set [17] is the set of participants that could possibly have sent a message. The receiver anonymity set [17] is the set of participants that could possibly have received a message. The anonymity set [17] is the entire set of participants. Ideally, the sender and receiver anonymity sets are equal to the anonymity set.

There has been a recent explosion in anonymity research: ten anonymity papers were published before 1996, over 125 papers have been published since (based on the extensive Freehaven anonymity bibliography). In addition, there have been numerous conferences with privacy and anonymity as the central theme. Therefore, for brevity reasons, we present only the four main anonymity techniques [12] here: mixes, re-routing, buses, and broadcasting. More examples that use these anonymity techniques can be found at [1].

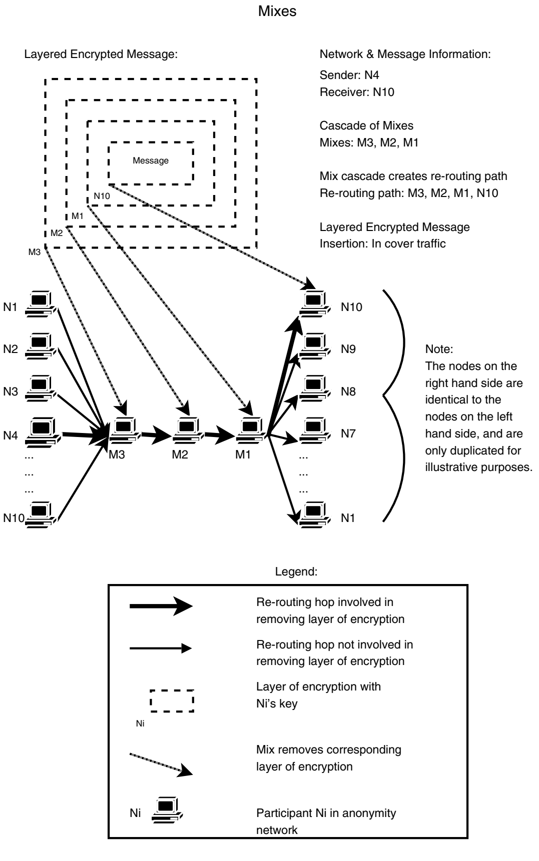
Mixes
Layered Encrypted Message:
Message
N10
M1
M2
M3
Network & Message Information:
Sender: N4
Receiver: N10
Cascade of Mixes
Mixes: M3, M2, M1
Mix cascade creates re-routing path
Re-routing path: M3, M2, M1, N10
Layered Encrypted Message
Insertion: In cover traffic
N1
N2
N3
N4
...
N10
M3
M2
M1
N10
N9
N8
N7
...
N1
Note:
The nodes on the right hand side are identical to the nodes on the left hand side, and are only duplicated for illustrative purposes.
Legend:
Re-routing hop involved in removing layer of encryption
Re-routing hop not involved in removing layer of encryption
Layer of encryption with Ni's key
Ni
Mix removes corresponding layer of encryption
Ni
Participant Ni in anonymity network

Figure 1: Mixes protocol overview.

In 1981, David Chaum started the field of anonymous communication by publishing the first paper on mixes [3]. The idea is to re-route an encrypted message through an intermediate node, called a mix. The purpose of the mix is to collect a batch of anonymous messages, and reorder the decrypted outputs randomly in order to obfuscate any input-output timing correlations. Uniform message size and layered encryption are used to prevent correlations of input-output size and content. Constant traffic from every sender to every receiver is used to prevent counting attacks on the number of inputs and outputs at each mix in a low traffic network. A cascade of mixes is used as the re-routing path, so that as long as one of the mixes obfuscates its input-output correlations, anonymity is maintained. The problem with mixes is that the constant traffic and known network topology requirements are not scalable. Despite this, they are used for applications that can tolerate high latency, such as the anonymous re-mailer *Mixminion* [7], because of their strong anonymity guarantees.

Figure 1 shows an example of a three mix cascade (M3, M2, and M1). There are ten senders, who are also receivers. N4 sends a layered encrypted message to the receiver N10, by inserting the message into the cover traffic through the mix cascade. Each mix in the cascade, as well as the receiver, peels away its respective layer of encryption.

Due to the lack of scalability, but a need for anonymous communication, mixes were reduced down to re-routing techniques [8]. Re-routing techniques eliminate the cover traffic requirement, and reduce the expensive cryptographic computation overhead. This yields a low-latency anonymous communication scheme, at the cost of being susceptible to multiple attacks. Interactive applications, such as web browsing and ssh, can tolerate only a low-latency overhead in order to be practical. The caveat is that an adversary with enough resources can defeat anonymity. For example, a global view of the anonymity network allows a trivial timing correlation attack that traces a message from the sender to the receiver.

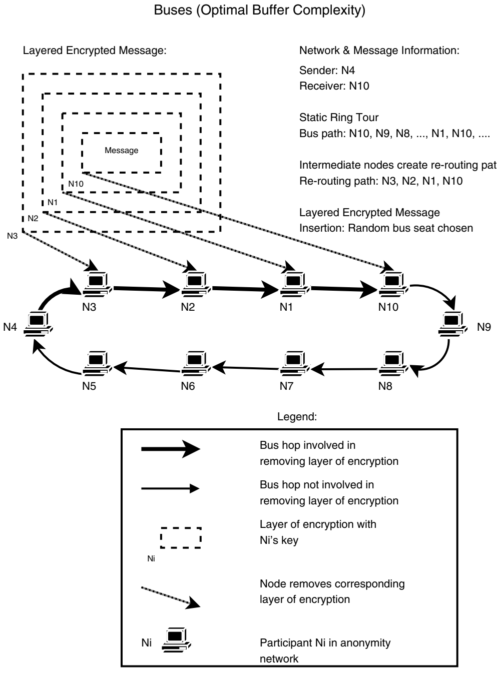
Buses (Optimal Buffer Complexity)
Layered Encrypted Message:
Message
N10
N1
N2
N3
Network & Message Information:
Sender: N4
Receiver: N10
Static Ring Tour
Bus path: N10, N9, N8, ..., N1, N10, ...
Intermediate nodes create re-routing path
Re-routing path: N3, N2, N1, N10
Layered Encrypted Message
Insertion: Random bus seat chosen
N3
N2
N1
N10
N4
N9
N5
N6
N7
N8
Legend:
Bus hop involved in removing layer of encryption
Bus hop not involved in removing layer of encryption
Layer of encryption with Ni's key
Ni
Node removes corresponding layer of encryption
Ni
Participant Ni in anonymity network

Figure 2: Buses optimal buffer complexity protocol overview.

Buses is a relatively new anonymity technique, published by Beimel and Dolev in 2003 [2], that uses the metaphor of a city bus to deliver anonymous messages contained in the seats of the bus. Semantically secure layered encryption is used to prevent an adversary from determining if a decrypted bus seat contains random data, or a multiply encrypted message. The bus traverses the anonymous network, and participants decrypt the bus, extract valid messages, replace them with random data, and randomly insert layered encrypted messages to be delivered anonymously. The replacement scheme and semantically secure encryption hides the event of when a sender sends a message. The future of buses is promising, since it can provide strong anonymity like mixes, but does not require costly cover traffic. Currently, it is a medium-latency anonymous communication scheme. Research is being done to reduce it to a low-latency anonymous communication scheme, without sacrificing its strong anonymity.

An example of a ten node buses anonymity network is depicted in Figure 2. The ten peers are both senders and receivers. N4 sends a layered encrypted message to N10 on the bus, via the intermediate nodes in the bus path (N3, N2, and N1). Each intermediate node, as well as the receiver, peels away its respective layer of encryption.

There is another technique, also introduced by Chaum, that uses broadcasting for anonymous communication [4]. A participant sends a message by broadcasting a specially-encoded version of the message to all other participants. However, costly contention issues where multiple participants try to send a message at the same time must be monitored carefully and resolved. This anonymity technique has fallen out of favour because broadcasting to all participants is not scalable.

3. ANONYMOUS COMMUNICATION ATTACKS

In order to analyse the security of an anonymous communication scheme, a threat model is used, which typically consists of attackers with varying resources [13]. An eavesdropper can eavesdrop on messages exchanged by a subset of the participants. A global eavesdropper can eavesdrop on all the messages exchanged by all the participants. A passive adversary can corrupt one or more participants, and observe messages being sent and received, re-routing tables, the participant's private keys, and so on for the corrupted participant. An active adversary has all of the powers that a passive adversary has, in addition to the ability to delete, add, or modify messages. The goal of an eavesdropper or passive adversary is to observe events and decrease anonymity. The objective of an active adversary is to create events to further decrease anonymity, in addition to observing events.

There are numerous anonymity attacks that exist [13]. For example, an attacker can try to determine the re-routing path via size, timing, content, or tagging of a message as it is re-routed through the network. A replay attack could replay a message to look for an intersection of common events, to trace a message to its receiver. A traceback attack could corrupt the anonymous routing tables on the reverse path from the receiver. Alternatively, key loggers on a sender's machine could easily defeat sender anonymity.

The best method to defend against anonymity attacks is to use an anonymity technique with minimal observable events, reducing an attacker's advantage. Clearly, mixes is a better choice than re-routing, if high latency and lack of scalability can be tolerated. However, buses are a better choice than mixes because they provide just as good anonymity as mixes, but are a medium-latency anonymity technique [12].

4. DEPLOYMENT OF ANONYMOUS MALWARE

It is only a matter of time until anonymous malware is seen in the wild because of the untraceability advantages to the malware author and monetary advantages to the groups that use it for illicit communication. Before describing the deployment of anonymous malware, however, it is prudent to evaluate whether it is ethically acceptable to present such information. We feel that it is ethically acceptable if, and only if, defences against anonymous malware are presented. In addition, we present only enough details of the anonymous malware required to present concrete defences and counterattacks.

To narrow down what type of anonymous communication technique would be deployed with malware, three characteristics of the anonymity technique are required.

1. The anonymity technique must be able to handle a dynamic membership and scale well. Otherwise, dynamic connectivity, disinfection of infected nodes, and the size of the Internet, would not provide acceptable scalability with respect to performance.
2. The sender and receiver of a message should be able to be any node in the anonymous network. Otherwise, dynamic membership could easily prevent communication if either all of the send points are down, or all of the receive points are down.
3. The anonymity technique should provide strong anonymity. Otherwise, there could be severe repercussions of being caught and prosecuted. The severity of repercussions is currently on the rise [22].

These requirements narrow the acceptable anonymity techniques to mixes and buses. Strong anonymity is required, so the re-routing technique is not acceptable. The anonymity technique should also scale well, so the broadcasting technique is eliminated. Clearly, a peer-to-peer (P2P) design is a good choice since the anonymity technique must be able to handle dynamic membership and allow any node to send or receive a message. There are already P2P mixing techniques implemented, such as *Tarzan*. However, the buses technique is a more suitable candidate since it does not require the entire network topology to be known. A sent message needs only to know the buses to take, and their corresponding transfer stations, along the path to the receiver. Furthermore, the scalability of buses is good since it does not require constant cover traffic.

Both anonymity techniques, buses and mixes, have common observable events. Both use end-to-end layered encryption, and both allow a node/mix to identify the predecessor by observing who sent a message. In addition, both identify other participants by observing who the recipient is of a forwarded message/bus.

To allow concrete defences to be presented, the process of setting up an anonymous malware network using buses in a covert manner is presented next. The general process is to construct a hit-list [20], covertly create a seed of corrupted hosts, propagate the anonymous malware by scanning and infecting additional Internet hosts, and stop propagation in a controlled way once the anonymity network is sufficiently large.

The size of an anonymity network is an important factor, because a network that is too small will provide little in the way of cover. With buses, the size of the anonymity network is equal to the sender and receiver anonymity sets. The spread of anonymous malware can be halted by the author anonymously controlling the propagation, eventually sending a stop propagation message once the anonymity network is large enough. Alternately, the malware can model its own spread, ceasing further propagation activity when it estimates that enough machines have been compromised [23].

To overcome the slow start in propagation, a hit-list should be constructed, and a subset of the hit-list (the seed) should be infected. The hit-list identifies which Internet hosts are believed to be vulnerable to the attack vector. This can be accomplished by scanning the Internet with a botnet, or a pre-existing anonymous malware network. Then, a subset of the hit-list would be infected in order to produce a large enough seed. The hit-list would, by its nature, also document a number of likely points of access to the malware-created anonymity network for would-be senders and receivers.

Once a large enough seed is created, the propagation would begin. This would consist of the hit-list left over after seeding being divided among the seed, and further divided among new targets infected by scanning. After the hit-list is exhausted, scanning would have a preference for local subnets (e.g. class C addresses) over external networks. If one node in the subnet is infected, chances are that additional nodes in the subnet are under the same administrative control and are vulnerable to the same attack vector.

To make the propagation more covert and resistant to traffic analysis, the anonymous malware author could use a bus that traverses the infected nodes. Each infected node would require a ‘token’ in order to perform its next scan and ensuing infection attempt. The author could remotely control the placement policy of these scanning tokens on the bus, which would deliver the scanning tokens to the nodes. Ideally, the token placement policy results in a diversified scanning base that spreads its scans out over time sufficiently, so that any detected scans are not correlated. In addition, the bus could anonymously gather anonymity network information such as membership, public keys, approximation of the anonymity network size, etc. This information would be hidden with public key encryption using the public key from a public-private key pair that the malware author generates.

The routing of the bus would require additional instrumentation, above and beyond the buses anonymity technique. Let the set of nodes that are infected be denoted as the participants. When a participant is initially infected, it records the IP address and key of its predecessor (making the anonymity network weakly connected), and also receives from the predecessor a random subset of the predecessor’s local topology with the corresponding keys. If the local topologies are weakly connected, the bus can tour the entire network with a random walk.

Due to infected nodes becoming disinfected or disconnected, a modified version of ‘gossiping’ is needed to maintain the weakly connected anonymity network. Our solution is based upon a variant of gossiping in *Tarzan* [10]. During the initialization phase, a node sends its topology information, including the corresponding public keys, to another random node. The recipient augments its topology and public keys with the topology and public keys received from the neighbour. Nodes in the local topology and their corresponding public keys are lazily pruned when a node does not respond. However, the anonymous malware would require the topology, with the corresponding keys, to be sent covertly and be bounded in size to be efficient. To accomplish this, a node would propagate a random subset of its local topology and corresponding keys, bounded in size, via an anonymous message on the bus to a random recipient. In addition, it is beneficial for the anonymity network to be weakly connected, instead of strongly connected, so that a random walk is more efficient. As a result, a recipient could randomly choose a subset of the received topology and key information and augment its own local topology and corresponding keys up to a predefined maximum size.

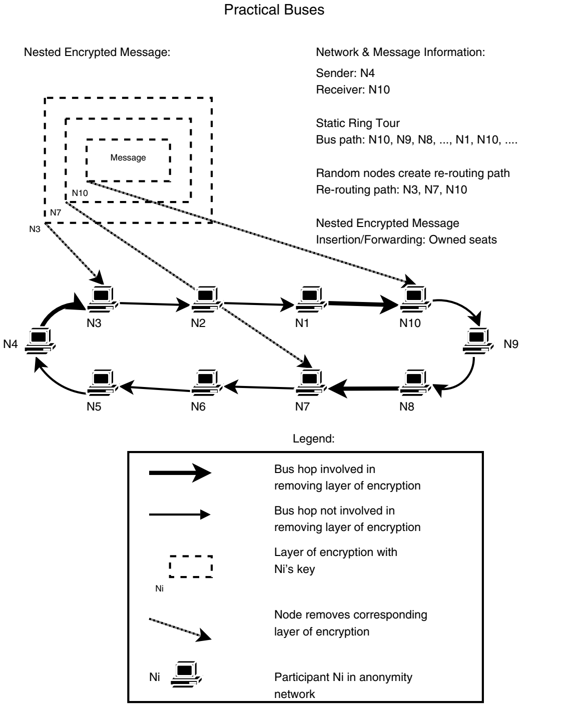
Practical Buses
Nested Encrypted Message:
Network & Message Information:
Sender: N4
Receiver: N10
Static Ring Tour
Bus path: N10, N9, N8, ..., N1, N10, ...
Random nodes create re-routing path
Re-routing path: N3, N7, N10
Nested Encrypted Message
Insertion/Forwarding: Owned seats
Message
N10
N7
N3
N3
N2
N1
N10
N4
N9
N5
N6
N7
N8
Legend:
Bus hop involved in removing layer of encryption
Bus hop not involved in removing layer of encryption
Layer of encryption with Ni's key
Ni
Node removes corresponding layer of encryption
Ni
Participant Ni in anonymity network

Figure 3: Practical buses protocol overview.

Additional mechanisms to send and receive a message are also needed, as in the Practical Buses Protocol [12]. A participant selects an indirection path, which is a random list of bounded length from the nodes in the local topology, appended with the receiver of the message. Then, the message is encrypted recursively on the reverse of the indirection path and placed on the bus. When an indirection node decrypts a valid seat to forward, the corresponding decryption is placed on the bus. Eventually, all the layers of encryption are peeled away and the receiver receives the message.

An example of a ten-node practical buses anonymity network is given in Figure 3. Again, the ten peers are both senders and receivers. N4 sends a nested encrypted message to N10, via the randomly chosen indirection nodes (N3 and N7). Each indirection node, as well as the receiver, peels away its respective layer of encryption.

In addition, instrumentation is required to hide the insertion of seats, and for a node to know when a seat it has sent has been received by the intended node. These details are not needed to explain the defences, but can be found in [12], where it is also shown that the protocol is secure and the attacks to defeat anonymity require a well resourced attacker. The sender anonymity set and receiver anonymity set can consist of the entire anonymity set.

If an infected node does not hide the fact that it is sending anonymous traffic, the infection is easily identified by monitoring the traffic sent and received by the infected node. As a result, it is essential that an infected node hides the observable event of when it is sending or receiving the bus. The caveat is that the traffic must appear normal, and may potentially pass through a firewall. In the event that the port is filtered by an egress firewall, then a non-filtered port must be used. For egress firewalls that are fine-grained enough to block individual applications as well as ports, then the corresponding application that is accepted for that port must be corrupted. In addition, anonymous traffic must be sent when the port's regular traffic is sent. Otherwise, an observant user will identify the temporal anomaly.

5. ANONYMOUS MALWARE DEFENCES

How can anonymous malware be defended against? Traditional malware defences that identify the initial scanning and infection of a node via the attack vector are an excellent first defence. These defences can consist of anti-virus software, firewalls, and regular OS and software patching. However, these defences can be subverted:

- Well-designed malware can be missed by anti-virus software. The malware can covertly spread slowly, create no noticeable events, and not exhibit behaviour characteristics used by anti-virus heuristics to identify malware.
- Firewalls do not provide an absolute defence. The attacker only needs to corrupt an application which is already allowed to communicate through the firewall.
- Regular OS and software patching is important, but still leaves a time where there is a window of opportunity for the attacker. This time window is further increased for corporate networks, where IT policy may require that any new patch is tested first before deployment.

Clearly, a secondary defence is needed that is designed with anonymous malware communication in mind. The novel technique we suggest is to monitor traffic for encrypted data blocks being sent or received. This monitoring would alert the user when encrypted data is being sent/received, which application file it is being sent/received by, and the destination/source address. Similar to the lock icon in *IE* for an encrypted connection, a lock icon could be placed in the task bar. It could be bundled into firewall software, and the traffic could be delayed with a pop-up prompting the user to deny or allow the traffic. Rules could be created to improve usability. In the event that a previously created rule has its corresponding application infected, the lock icon plays an important role. If the user is not expecting encrypted traffic and the lock appears, the anomaly is detected and the firewall log can be analysed.

The key to identifying encrypted data is that a good encryption scheme must produce 'random' output, otherwise it is easily broken. For example, 189 statistical randomness tests were used to evaluate the finalists of the AES competition [19]. As a result, monitoring of traffic could simply test different block sizes for randomness [14]. If the randomness surpasses a threshold for the particular application, or uses a previously unknown source/destination address, it could result in a query being sent to the user via an allow/deny pop-up.

The monitoring of encrypted traffic would be a good secondary defence for both mixes and buses. It would also be a good countermeasure for malware that calls home with mined confidential data that is encrypted.

As a countermeasure, malware could insert redundancy in order to prevent detection. However, this arms race is quickly defeated since any non-random data creates an easily identifiable signature.

A tertiary defence is to watch for unexpected CPU usage. Decryption is CPU-intensive. However, this alone would create too many false positives.

There are two caveats of the secondary and tertiary defences. First, user education is paramount. Otherwise, the user will not be able to interpret if their machine is infected by anonymous malware. Second, the pervasiveness of encrypted communications is increasing. This makes it more difficult for the user to determine whether encrypted traffic is legitimate or not.

The accuracy of detection for the secondary and tertiary defences could be improved by using an expert system that watches for correlations between the receipt of encrypted data, CPU usage, and the sending of encrypted data. The expert system could filter out events that have a low likelihood of being generated by anonymous malware.

6. COUNTERATTACK TECHNIQUES

If a node is discovered to be infected by anonymous malware, it should be disinfected. However, immediate disinfection is not a good neighbour policy, since an infected node can be watched to determine other infected nodes, similar in spirit to some classic episodes [5, 21].

The purpose of delaying disinfection is to learn about other infected nodes in the anonymous network. The anonymity attack techniques that exist in the literature use the collection and creation of observable events to defeat sender and/or receiver anonymity. However, this turns out to be much more complex than learning anonymity network membership. A good neighbour watches who forwards anonymous messages to it, and the recipients of the anonymous messages that it forwards. The good neighbour can then inform any other neighbours that it suspects of being infected with anonymous malware. Any suspected neighbours should be provided with enough details to ascertain whether they are infected. These details could include the application the malware corrupts, the addresses it uses to communicate (e.g. port), observed anonymous messages it forwarded to the good neighbour, etc. Of course, anti-virus vendors should be made aware so that traditional defences are able to stop the anonymous malware before infection, if at all possible.

To prevent the node from contributing to the anonymity network, the node should corrupt any anonymous traffic. Any anonymous messages it forwards should be XORed with a random bit sequence. This random bit sequence could be generated by any stream cipher with a random initialization vector, such as SEAL [15]. This corrupted encrypted traffic would not decrypt properly, and such decryption anomalies would be considered normal if any cover traffic is used. Even if acknowledgments are used by the anonymous communication scheme, it would take time for the anonymity network to confidently ascertain and prune an infected node that is discovered and destroying messages.

7. CONCLUSION

Anonymous malware will appear in the future, as it is in the best interest of malware authors to prevent themselves from being identified. There are similar advantages to users of malware-constructed anonymity networks. For example, spyware could send harvested credit card information through an anonymity network that is used for coordinating organized crime. The buses anonymity technique is a good candidate to be used with anonymous malware, because it is a P2P strong anonymous communication scheme with a medium latency overhead that scales well.

The initial defences against anonymous malware are the traditional ones: anti-virus software, firewalls, regular patching. New defences monitor the reception of encrypted traffic, the overhead of decrypting encrypted traffic, and the sending of encrypted traffic in order to detect unexpected activity. The key is the correlation between these events, and the unexpected sender and receiver addresses of the encrypted data. An expert system could be used to filter out superfluous reports to the user. As always, user education is paramount, because ultimately the user will be called upon to assess whether or not questionable traffic is legitimate.

Once a node is ascertained to be infected, it can be restored to an uninfected state. However, it is important that an infected node be a good neighbour and notify other nodes that it suspects to be infected. Continuing to participate in a malicious anonymity network does not imply faithfully forwarding messages. A good neighbour will hinder the anonymous communication by XORing all anonymous messages with a random sequence of bytes.

The combination of these defences will give a good head start to preventing the anonymous flow of malware, illicit communication, and mined confidential information, as well as identifying the computers used by surreptitiously-established anonymity networks.

8. ACKNOWLEDGMENTS

Carey Williamson commented on a draft of this paper. The research of both authors is supported in part by the Natural Sciences and Engineering Research Council of Canada; the first author's research is also supported by the Alberta Ingenuity Fund and iCORE.

REFERENCES

[1] Anonymity bibliography, 2005. http://www.freehaven.net/anonbib/.

[2] Beimel, A. and Dolev, S., 'Buses for anonymous message delivery', *Journal of Cryptology*, 16(1):25–39, 2003.

[3] Chaum, D., 'Untraceable electronic mail, return addresses and digital pseudonyms', *Communications of the ACM*, 24(2):84–88, 1981.

[4] Chaum, D., 'The dining cryptographer's problem: Unconditional sender and recipient untraceability', *Journal of Cryptology*, 1(1):65–75, 1988.

[5] Cheswick, B., 'An evening with Berferd in which a cracker is lured, endured, and studied', in *Proceedings of the Usenix Winter '92 Conference*, pp.163–174, 1992.

[6] Claessens, J., Preneel, B. and Vandewalle J., 'Solutions for anonymous communication on the Internet', in *Proceedings of the International Carnahan Conference on Security Technology*, pp.298–303, IEEE, 1999.

[7] Danezis, G., Dingledine, R. and Mathewson, N., 'Mixminion: Design of a type III anonymous remailer protocol', in *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, May 2003.

[8] Dingledine, R., Mathewson, N. and Syverson, P., 'Tor: The second-generation onion router', in *Proceedings of the 13th USENIX Security Symposium*, August 2004.

[9] Filiol, E., 'Strong cryptography armoured computer viruses forbidding code analysis: The Bradley virus', in *Proceedings of the 14th Annual EICAR Conference*, pp.216–227, 2005.

[10] Freedman, M. and Morris, R., 'Tarzan: A peer-to-peer anonymizing network layer', in *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS 2002)*, Washington, DC, November 2002.

[11] Guan, Y., Fu, X., Bettati, R. and Zhoa, W., 'An optimal strategy for anonymous communication protocols', in *Proceedings of 22nd International Conference on Distributed Computing Systems*, pp.257–266, IEEE, 2002.

[12] Hirt, A., 'A practical buses protocol for anonymous network communication', Master's thesis, University of Calgary, Calgary, Alberta, August 2004.

[13] Hirt, A., Jacobson, M. and Williamson, C., 'Survey and analysis of anonymous communication schemes', *ACM Computing Surveys*, 2004, submitted for publication.

[14] Knuth, D. E., *The Art of Computer Programming, Volume 2: Seminumerical Algorithms,* Addison-Wesley, 3rd edition, 1998.

[15] Menezes, A., van Oorschot, P. and Vanstone, S., *Handbook of Applied Cryptography*, CRC Press, Boca Raton, Florida, 2001.

[16] *FOX News*, 'U.N. whistleblower to testify on oil-for-food', March 2005. http://www.foxnews.com/story/0,2933,150678,00.html.

[17] Pfitzmann, A. and Köhntopp, M., 'Anonymity, unobservability, and pseudonymity: A proposal for terminology'. Draft, version 0.17, July 2004.

[18] Pfitzmann, A. and Waidner, M., 'Networks without user observability', *Computers & Security*, 2(6):158–166, 1987.

[19] Sotto, J. and Bassaham, L., 'Randomness testing of the advanced encryption standard finalist

candidates', Technical report, National Institute of Standards and Technologies, 2000.

[20] Staniford, S., Paxson, V. and Weaver, N., 'How to 0wn the Internet in your spare time', in *Proceedings of the 11th USENIX Security Symposium*, 2002.

[21] Stoll, C., 'Stalking the wily hacker', *Communications of the ACM*, 31(5):484–497, 1988.

[22] USA PATRIOT ACT, October 2001. http://www.epic.org/privacy/terrorism/hr3162.html.

[23] Vogt, T., 'Simulating and optimizing worm propagation algorithms', 2003. http://downloads.securityfocus.com/library/WormPropagation.pdf.